A beginners introduction and guide to RMark

Description

The RMark library is a collection of R functions that can be used as an interface to MARK for analysis of capture-recapture data.

Details

The library contains various functions that import/export capture data, build capture-recapture models, run the FORTRAN program MARK.EXE, and extract and display output. Program MARK has its own user interface; however, model development can be rather tedious and error-prone because the parameter structure and design matrix are created by hand. This interface in R was created to use the formula and design matrix functions in R to ease model development and reduce errors. This R interface has the following advantages:

- Uses model notation to create design matrices rather than designing them by hand in MARK or in EXCEL, which makes model development faster and more reliable. All-different PIMS are automatically created for each group (if any).

- Allows models based on group (factor variables) and individual covariates with groups created on the fly. Age, cohort, group and time variables are pre-defined for use in formulas.

- Both real and beta labels are automatically added for easy output interpretation.

- Input, output and specific results (eg parameter estimates, AICc etc) are stored in an R object where they can be manipulated as deemed useful (eg plotting, further calculations, simulation etc).

- Parameter estimates can be displayed in triangular PIM format (if appropriate) for ease of interpretation.

- Easy setup of batch jobs and the calls to the R functions “document” the model specifications and allow models to be easily reproduced or re-run if data are changed.

- Covariate-specific estimates of real parameters can be computed within R without re-running the analysis.

The following are the MARK capture-recapture models that are currently supported:

model	Selection in MARK
CJS	Recaptures only
Recovery	Recoveries only
Burnham	Both(Burnham)
Barker	Both(Barker)
Pradel	Pradel recruitment only
Pradsen	Pradel survival and seniority

`Pradlambda`	Pradel survival and lambda
`Pradrec`	Pradel survival and recruitment
`LinkBarker`	Available only in change data type as Link-Barker
`Closed`	Closed - no heterogeneity
`HetClosed`	Closed – heterogeneity
`FullHet`	Closed – full heterogeneity
`Huggins`	Huggins – no heterogeneity
`HugHet`	Huggins - heterogeneity
`HugFullHet`	Huggins – full heterogeneity
`POPAN`	POPAN
`Known`	Known - known fate data (e.g, radio-tracking)
`Multistrata`	Multistrata - CJS model with strata

There is one limitation of this interface beyond the obvious that it does not currently handle all of the models and capabilities in MARK. All models in this interface are developed via a design matrix approach rather than coding the model structure via parameter index matrices (PIMS). This excludes the use of the sin link in MARK and it has implications for MARK's ability to count the number of identifiable parameters (see `dipper` for an example).

Before you begin, you must have installed MARK (http://www.cnr.colostate.edu/~gwhite/mark/mark.htm) on your computer or at least have a current copy of MARK.EXE. As long as you selected the default location for the MARK install (c:/Program Files/Mark), the `RMark` library will be able to find it. If for some reason, you chose to install it in a different location, see the note section in `mark` for instructions on setting the variable MarkPath to specify the path. In addition to installing MARK, you must have installed the `RMark` library into the R library directory. Once done with those tasks, run R and enter library(RMark) (or put it in your .First function) to attach the library of functions.

The following is a categorical listing of the functions in the library with a link to the help for each function. To start, read the help for functions `import.chdata` and `mark` to learn how to import your data and fit a simple model. The text files for the examples shown in `import.chdata` are in the subdirectory RMarkdata within the R Library directory. Next look at the example data sets and analyses `dipper`, `edwards.eberhardt`, and `example.data`. After you see the structure of the examples and the use of functions to fit a series of analyses, explore the remaining functions under Model Fitting, Batch Analyses, Model Selection and Summary and Display. If your data and models contain individual covariates, read the section on Real Parameter Computation to learn how to compute estimates of real parameters at various covariate values.

Input/Output data & results

`import.chdata`, `export.chdata`, `read.mark.binary`, `extract.mark.output`

Model Fitting

`mark`, `process.data`, `make.design.data`, `add.design.data`, `make.mark.model`, `run.mark.model` `merge.occasion.data`

Batch analyses with functions

`run.models`, `collect.models`, `create.model.list`, `mark.wrapper`

Summary and display

`summary.mark`, `print.mark`, `get.real`, `compute.real` `print.summary.mark`, `print.mark.list`

Model Selection

`adjust.chat`, `adjust.parameter.count`, `model.table`

Real Parameter computation

`find.covariates`, `fill.covariates`, `compute.real`

Utility and internal functions

`collect.model.names`, `compute.design.data`, `extract.mark.output`, `inverse.link`, `deriv.inverse.link`, `setup.model`, `setup.parameters`, `valid.parameters` `cleanup`

For examples, see `dipper` for CJS and POPAN, see `example.data` for CJS with multiple grouping variables, see `edwards.eberhardt` for various closed-capture models, see `mstrata` for Multistrata, and see `Blackduck` for known fate. The latter two are examples of the use of `mark.wrapper` for a shortcut approach to creating a series of models.

Author(s)

Jeff Laake

References

MARK: Dr. Gary White, Department of Fishery and Wildlife Biology, Colorado State University, Fort Collins, Colorado, USA http://www.cnr.colostate.edu/~gwhite/mark/mark.htm

---

[Package *RMark* version 1.3 Index]